Generating Tessellation Files From C++ Code

1. Introduction

We provide a C++ code that can generate a tessellation file from a given set of input information about nodes, faces, edges, etc.

2. Requirements

You would need a g++ compiler version 13.0.0 or later to build the C++ code (earlier versions supporting vector, string, and the specified header files in the code would also work). In addition to this, you just need to have any text editor installed on your system.

3. Inputs to the code

2.1 2-D tessellation

The CPP files would need three csv files as input. All the csv files should not have row names and column names. They should directly start with the desired information.
This is how a typical node.csv file should look:

node

1	**1**	**1**	**0**
2	2	0	0
3	3	1	0
4	2	2	0

Note that the first row is the information corresponding to the first node and the first column is the ID for all the nodes. Similar conventions should be followed for all other csv files.

These files are:

1. A csv file containing the information about the nodes. The schema for this file is given below. We have assumed weight to be zero.

Node ID	X coordinate	Y coordinate	Z coordinate	Weight

2. A csv file containing information about the edges. The edge direction would be from Node1 to Node2. The schema for this file is given below. We have assumed weight to be zero.

Edge ID	NodeID 1	NodeID 2	Weight

3. A csv file containing information about the facets and their binding edges given in counter-clockwise manner. The schema for this file is given below. Note that the total number of columns would depend on the number of binding edges provided in that row.

Facet ID	Number of binding edges	Binding edge ID 1	Binding edge ID 2	Binding edge 3	Binding edge 4

2.2 3-D tessellation

The CPP files would need four csv files as input. In addition to the three csv files on nodes, edges and faces, a file containing the polyhedron information should also be passed. This csv file contains information about the binding faces. The schema for this csv file is as follows:

Polyhedron ID	Number of binding faces	Binding face 1	Binding face 2	Binding face 3	Binding face 4	Binding face 5

Note that the total number of columns would depend on the number of binding faces provided in that row, similar to that in the case of facet information.

4. Usage

4.1 How to give inputs to the files?

If you want to generate a 2D tessellation, specify the names of the three files containing node, edge and facet information in the file named <filenames_2d.txt>. The names of the file should be separated by a space. All the csv files should be in the same directory as the c++ file.

If you want to generate a 3D tessellation, specify the names of the three files containing node, edge, facet and polyhedron information in the file named <filenames_3d.txt>. The names of the file should be separated by a space. All the csv files should be in the same directory as the c++ file.

4.2 Where and how to collect output tessellation files?

The contents for a 2D tessellation are written to a file named <output_2d.tess>.
The contents for a 3D tessellation are written to a file named <output_3d.tess>.

5. Limitations and side notes

The tesselation file we generate assumes a square domain for the 2D case while a cubic domain for the 3D case. The code is not capable of specifying any other type of outer domain.